Covert Channel Analysis-
Documentation Guidance

July 28, 2011

Contents

ACRONYMS and DEFINITIONS

Acronym	Definition
CCA	Covert Channel Analysis
CDS	Cross Domain Solution
DRD	Development Representation Documentation
DTLS	Descriptive Top-Level Specification
LLD	Low Level Design
MLS	Multilevel Security
SP	Security Policy
SRM	Shared Resource Matrix

1 Introduction

All CDS systems are based on a security model that is meant to enforce some sort of nondiscretionary access control policy. However, any implementation of a design is inevitably going to introduce unintended system behaviors which could be used by attackers to violate the system's security policy. In order to understand how the impact of such defects, a covert channel analysis (CCA) must be performed.

While most development artifacts for a CDS system are developed before and during the development process, the (CCA) may be executed both during and at the end of the development cycle. CCAs are typically most complete when executed using not only the system's design artifacts, but also the system's implementation and/or implementation representation. The assurance support provided by a CCA is then not only theoretical (i.e., the system is based on a sound design), but it also is concrete in that it provides analysis that is performed on implementation itself. Thus, a completed CCA document improves system assurance in many ways; it demonstrates the developer's knowledge of the system's security requirements, it provides a description of system behaviors, and it provides a means of defining techniques that can further reduce unintended information leakages. On the contrary, without a CCA document, evaluators and accreditors are left trusting that the CDS was implemented without error.

1.1 Document Goals

The purpose of this document is to provide guidance for the CCA document required for CDS systems. It is expected that CCA activities will vary substantially for each organization. This document is not intended to prescribe a particular method for CCAs, nor is it to specify a format for a CCA document. Rather, this document describes the following:

- Some key concepts fundemental to CCA.
- Requirements for the CCA document for a medium, medium-high, or high robustness CDS.
- Requirements for the CCA document that are specific to the different CDS classes (e.g., access, transfer, and multilevel).

Figure 1 depicts the relationship between the CCA document and the other documents within the DRD [6]. It is important to understand that not all covert channels are the result of implementation defects[5]. While many covert channels may be the result of defects in a system's implementation[1], some covert channels may be inherent to a system's very design. As a result, the CCA document should not be treated as a sort of defect report. Rather, the CCA

[1] In such cases, the covert channel should be treated as a defect and the developer should be required to eliminate the deficiency.

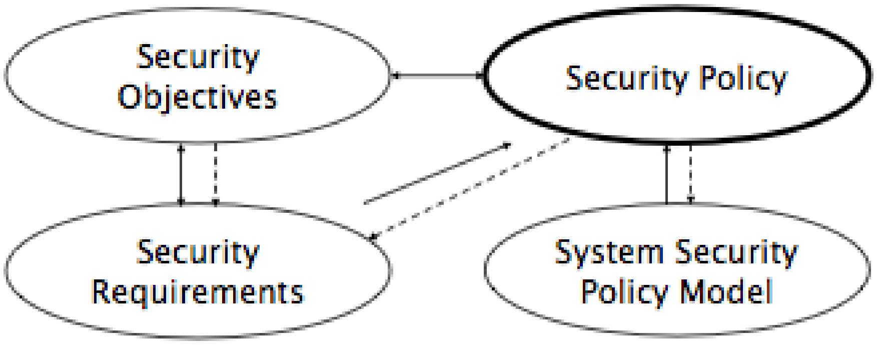

Figure 1: Relation of CCA Document to Other Documentation

document should be used a tool for better understanding how to manage the risks inherent to the use of CDS systems between domains that are otherwise meant to be separated.

2 Discussion

2.1 Terminology

While a full discussion regarding covert channels and covert channel analyses is beyond the scope of this document, some amount of discussion that covers some key concepts is in order. Below is a list of descriptions for some key terms that are fundemental to any investigation of covert channels. It is noteworthy that the following statements are descriptions, and as such are not meant to serve as formal definitions for the terms in question. However, concepts related to the terminology described below should be used in a CCA document.

- *covert channel*- Many definitions have been offered for the term 'covert channel'[3][4] . Essentially, a covert channel can be thought of as an information flow that uses system resources that are not designed or intended as data objects in a manner that violates the system's security policy[2].
- *bandwidth*- Any covert channel, like any other communication channel, has a limited bandwith [3]. The rate at which data may be transmitted via a covert channel is the channel's *bandwidth*. A channel's bandwidth can be described in theoretical terms (e.g., 'no more than one tenth of the overall data rate') or in measured terms (e.g., '1.5 kilobits per second').
- *shared resource matrix*- Any CDS is going to contain cases where a single resource is accessible, either for reading and/or for writing, by subsystems within the CDS that reside in separate security domains. Any such

[2] For more definitions, see section section 2.1 of [LPTSEC].

[3] Perhaps cite Shannon?

resource can be considered a *shared resource*. Simply put, a Shared Resource Matrix (SRM) can then be described as a table that associates shared resources with domains and their associated accesses for that resource[TODO: REF: Kemmerer].

- *covert storage channel*- A storage channel is a specific type of covert channel [4]. If a process sends or receives information using a shared object that is not meant as a communication channel, then that object can be considered a covert storage channel. For example, [TODO: Better example] were modulate the size of could be considered a covert storage channel.

- *covert timing channel*- A timing channel is a specific type of covert channel. If a process sends or receives information based on the time between events, then the time duration between the events can be considered a covert timing channel. For example, if the rate at which packets are sent over a network link are sent at a particular rate that has meaning to an external entity, then the packet transmission rate can be considered a covert timing channel.

- *aggregate covert channel*- An aggregate covert channel is a channel that uses a combination of more than one covert channels as a single communications channel. In some cases, several covert channels may be independently insignificant, but accessible in a manner that allows for a sender to use them in concert with each other to achieve useful amounts of bandwidth.

- *transient covert channel*- A transient covert channel is a channel that is available for a finite amount of usage, and then is no longer available. For example, if an entity can delete a file but not re-create it, then the deletion of the file could be considered a transient covert channel[TODO: change example to extracting audio from encrypted traffic].

- *side channel*- Most covert channels require two parties for communication: a transmitter and a receiver. However, in some cases, information can be inferred merely by observing the behavior of a system from the outside. In cases where a malicious transmitter is not necessary, the covert channel can be considered a *side channel*. For a simple example, if an adversary can extract information from a host based on the rate it which it sends normal, valid ssh packets, then the packet transmission rate could be considered a side channel.

2.2 Analysis and Abstraction

Covert channel analysis can be performed using any level of description of a system that is available[1] [2]. However, descriptions using only abstract system

[4] Differentiating between different types of covert channels is a challenging proposition. However, many times covert channels are described in terms of 'timing', 'storage', etc., and thus these terms will be used in this document.

descriptions are less likely to yield a useful CCA document than others that use both abstract and lower level descriptions. Thus, it is fair to say that CCA efforts that only use design artifacts are going to be less useful in developing a case for high system assurance than efforts that include analysis of the system's design, implementation, and implementation representation. This section briefly describes some different techniques for performing a CCA. It is noteworthy that the list below is not meant to be proscriptive, but rather is provided for informational purposes. Developers should feel free to perform the CCA at any and all appropriate levels of abstraction.

- *Design Analysis*- Several techniques exist for performing a CCA using design artifacts. Such techniques are useful in that they can quickly and efficiently identify shared resources that could *potentially* be exploited as a covert channel. Unfortunately, such analysis typically yields many "false positives", i.e., resources that could theoretically be used as covert channels, but cannot practically be exploited. Thus, while a CCA should use design analysis as a means for quickly determining potential covert channels, deeper analysis and/or measurement of system covert channels will ideally include other, lower level artifacts.

- *Source Analysis*- Source code provides a unique means for understanding the how a system's actual implementation effects the bandwidth of a covert channel associated with a shared system resource. For example, if a 32-bit integer is used as the storage mechanism for a shared resource that is polled once per second, then the actual bandwidth of the covert channel can be approximated with much more precision than could be surmised using design documentation exclusively.

- *System Testing*- While not ideal as a means for identifying covert channels, impementation testing is an optimal means for measuring covert channels that have been identified via other means. If design analysis yields knowledge regarding potential covert channels and source code analysis provides a means for bounding covert channels, the process of testing a system's implementation can provide hard data regarding the actual bandwidth of a covert channel. Indeed, even lacking source code and detailed design information, evaluators can test system implementations in an effort to demonstrate certain covert channels.

2.3 Impact of CDS Class

In theory, a CCA should be performed using standard techniques, regardless of the intent of the system. However, in most cases, the scope of analysis desired exceeds the amount of resources available for the analysis. As such, the developer and the evaluator are inevitably going to focus their efforts on the elements within the system that seem most pressing. Because the different classes of CDS typically have properties that are likely to increase risks in some aspects of their operation while reducing risks with otheres, a CCA will almost certainly vary in form based on the class of the CDS in question. This section describes some of the ways in which a CCA may differ based on the different CDS classes.

- *Access*- Access CDS systems are accessed directly by users and their applications. Also, access CDS systems are intended to enforce a null information flow policy. In order to even have a medium assurance that such a policy is being enforced in such an uncontrolled environment[5], some form of covert channel must be performed. In all cases, there is some shared resource within the system, for example, the keyboard, the display, etc. At a very minimum, a CCA must be performed for these components and/or subsystems based on the system's SRM.

- *Transfer*- Transfer CDS are designed to transmit information among disparate security domains. As such, any accreditor knows that there are some inherent covert channels existant in the system [6]. Also, transfer CDS systems are typically stored in tightly controlled facilities that are only accessible by a limited number of system administrators. These characteristics stand in stark contrast to those that characterize an access CDS. In response to these differences, a CCA for a transfer CDS should take these differences into account. For example, while a defect in the keyboard driver is less likely to result in data leakage[7], there are many more risks related to the fact that infromation initially transmitted on one domain is being retransmitted, in some form or fashion, in another domain. As such, a CCA and the resultant CCA document is likely to benefit for focusing more on ensuring that the development process took efforts manage and minimize covert channels in the system's exported data objects than on analyzing the characteristics of the keyboard driver.

- *Multilevel*- Multilevel systems can span a wide range of functionalities, and as such the CCA for a multilevel system will vary according to the functionality present in the CDS. In the case of true MLS systems, the solution has challenges similar to those accompanying access solutions. Namely, there are resources [8] that are shared by the different security domains, which can potentially be used to transmit and/or read information in a manner that violates the system security policy. As is the case with access solutions, some form of CCA for these shared resources must be executed even for medium assurance. In cases where there is

[5] While users, their processes, and their peripheral devices may be trusted to some degree (i.e., they are not necessarily malicious), it is difficult to ensure that these entities have not been corrupted and or tricked into acting on behalf of their adversaries.

[6] For example, the rate at which information is sent from high to low can always be used as a timing channel

[7] If I must explain this I will, but really!?

[8] Typically devices and the software that controls them, such as widnow managers, disk drives, etc.

non-MLS functionality in the mulitlevel system[9], the guidance applicable for transfer solutions applies; namely that more attention should be paid to externally visible data types. Becuase many multilevel systems have a large number of components that interact with multiple security domains, the level of effort associated with a CCA for multilevel systems will often be more complex than that of either transfer or access systems.

2.4 Impact of Robustness Level

This section describes the level of effort associated with a CCA for different levels of robustness

Rigor and coverage?

- Medium- design analysis
- Medium High- design analysis and testing
- High- design analysis, source code analysis, and testing
- Discuss scope of source code analysis, purpose, results
- use of srm at all levels, quality of srm at all levels.

3 Requirements

This section provides a list of the requirements for a CCA document.

CCA-1: The developer shall provide a CCA document.

CCA-2: The CCA document shall contain a shared resource matrix (SRM) for the system.

CCA-3: The CCA document shall characterize how items in the SRM are exposed to external interfaces.

CCA-4: The CCA document shall describe how data types transmitted to external entities are designed to manage covert storage channels.

CCA-5: For medium-high and high robustness systems, the CCA document shall provide guidance for managing covert channel bandwidth.

CCA-6: For high robustness systems, the CCA document shall provide formulas that are used to calculate the bandwidth associated with identified covert channels.

CCA-7: The CCA document shall include test results that provide measurements for identified covert channels.

CCA-8: For high robustness systems, the CCA document shall include a source code analysis [FIX THIS].

[9] For example, there may be a component that facilitates regrading/redaction from high to low

References

[1] Gerard Allwein. A qualitative framework for shannon information theories. Technical report, Naval Research Laboratory, August 2010.

[2] National Computer Security Center. A guide to understanding covert channel analysis in trusted systems. Technical report, National Security Agency, November 1993.

[3] CNSS. National information assurance (ia) glossary. Technical report, Committee on National Security Systems, April 2010.

[4] Butler W. Lampson. Protection. In *Communications of the ACM*, pages 613–615, Princeton, NJ, October 1973. Princeton University.

[5] C. E. Shannon. Communications theory of secrecy systems. Technical report, University of Illinois Press, 1993.

[6] HR CDS TT. Development representation documentation guidance. Technical report, Unified Cross Domain Management Office, June 2011.